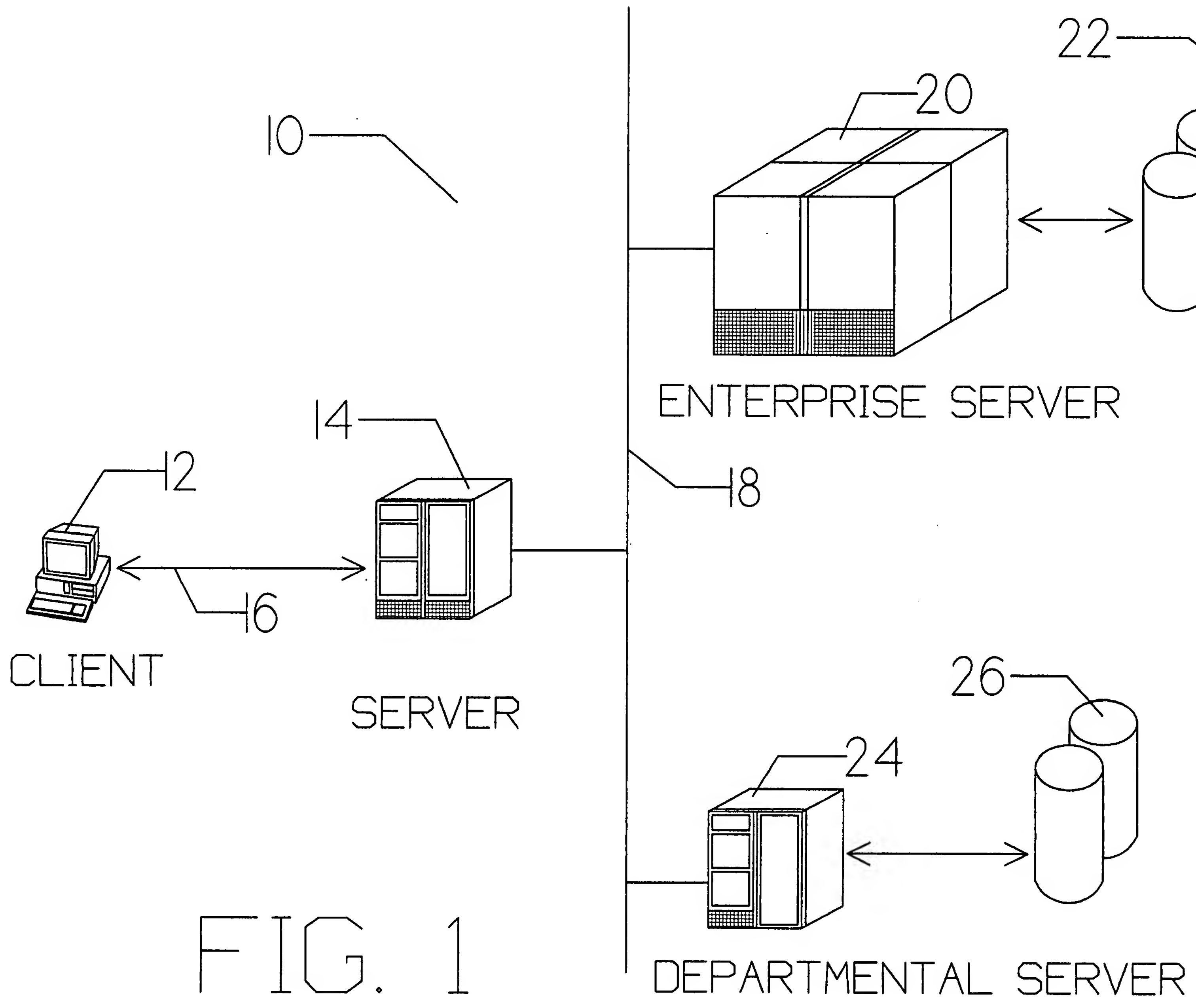

10
12
CLIENT
14
16
SERVER
18
20
22
ENTERPRISE SERVER
24
26
DEPARTMENTAL SERVER

FIG. 1

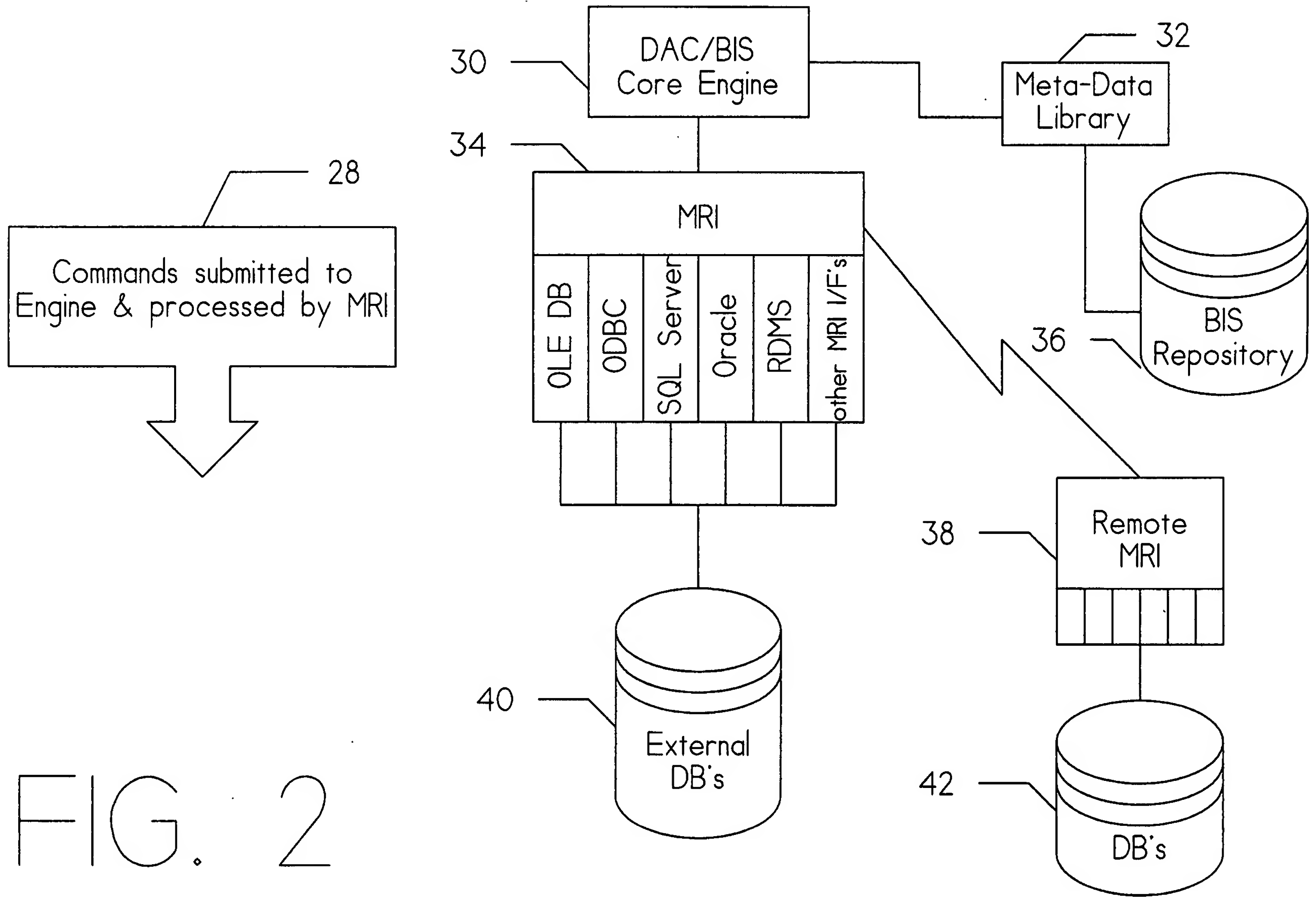

28
Commands submitted to Engine & processed by MRI
30
DAC/BIS Core Engine
32
Meta-Data Library
34
MRI
OLE DB
ODBC
SQL Server
Oracle
RDMS
other MRI I/F's
36
BIS Repository
38
Remote MRI
40
External DB's
42
DB's

FIG. 2

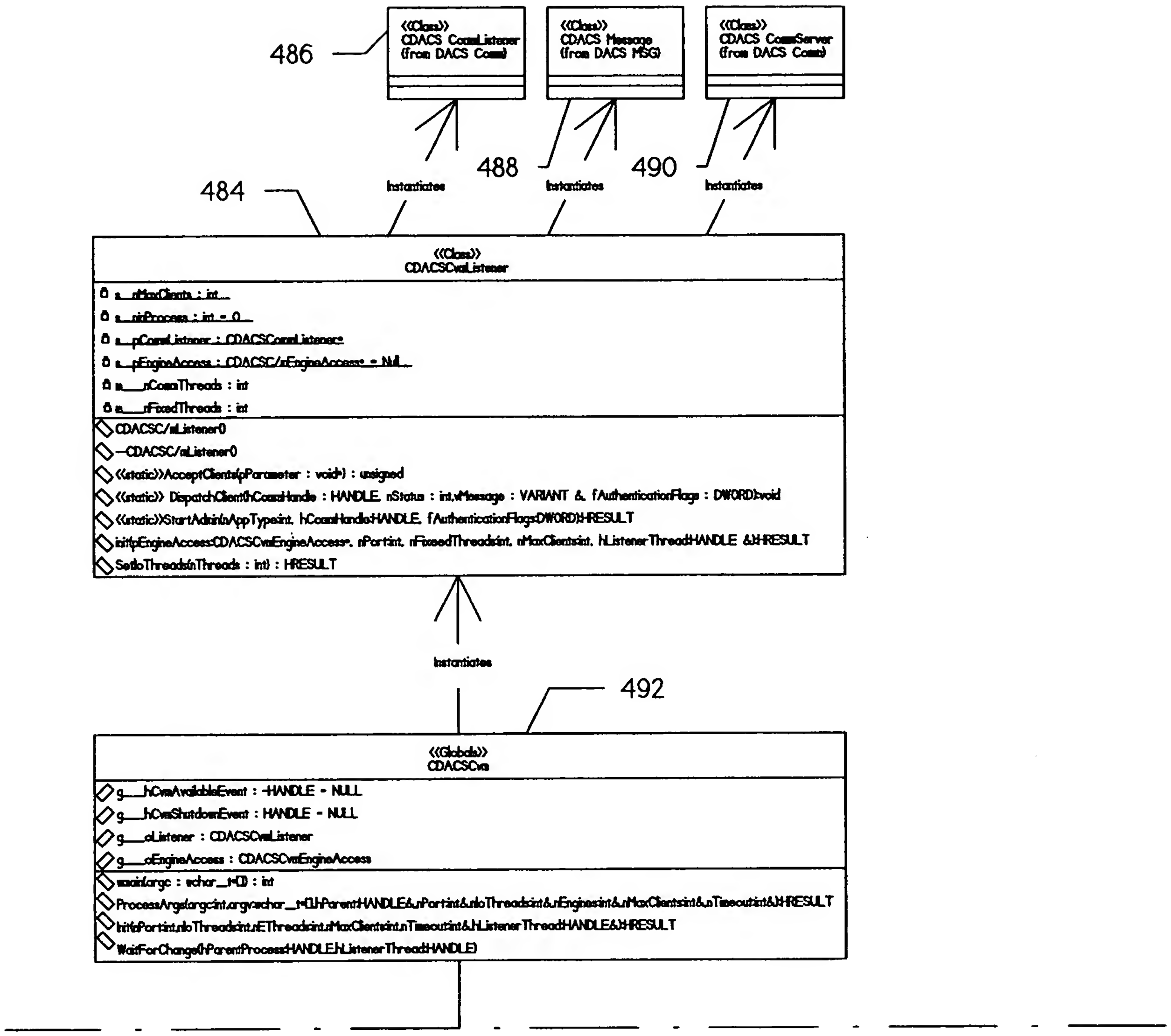

486
<<Class>>
CDACS CommListener
(from DACS Comm)
<<Class>>
CDACS Message
(from DACS MSG)
<<Class>>
CDACS CommServer
(from DACS Comm)
488
490
484
Instantiates
Instantiates
Instantiates
<<Class>>
CDACSCvmListener
s_nMaxClients : int
s_nInProcess : int = 0
s_pCommListener : CDACSCommListener*
s_pEngineAccess : CDACSCvmEngineAccess* = Null
m_nCommThreads : int
m_nFixedThreads : int
CDACSCvmListener()
~CDACSCvmListener()
<<static>>AcceptClients(pParameter : void*) : unsigned
<<static>> DispatchClient(hCommHandle : HANDLE, nStatus : int,vMessage : VARIANT &, fAuthenticationFlags : DWORD):void
<<static>>StartAdmin(nAppType:int, hCommHandle:HANDLE, fAuthenticationFlags:DWORD):HRESULT
init(pEngineAccess:CDACSCvmEngineAccess*, nPort:int, nFixedThreads:int, nMaxClients:int, hListenerThread:HANDLE &):HRESULT
SetIoThreads(nThreads : int) : HRESULT
Instantiates
492
<<Globals>>
CDACSCvm
g_hCvmAvailableEvent : HANDLE = NULL
g_hCvmShutdownEvent : HANDLE = NULL
g_oListener : CDACSCvmListener
g_oEngineAccess : CDACSCvmEngineAccess
wmain(argc : wchar_t*[]) : int
ProcessArgs(argc:int,argv:wchar_t*[],hParent:HANDLE&,nPort:int&,nIoThreads:int&,nEngines:int&,nMaxClients:int&,nTimeout:int&):HRESULT
Init(nPort:int,nIoThreads:int,nEThreads:int,nMaxClients:int,nTimeout:int&,hListenerThread:HANDLE&):HRESULT
WaitForChange(hParentProcess:HANDLE,hListenerThread:HANDLE)

FIG. 3A

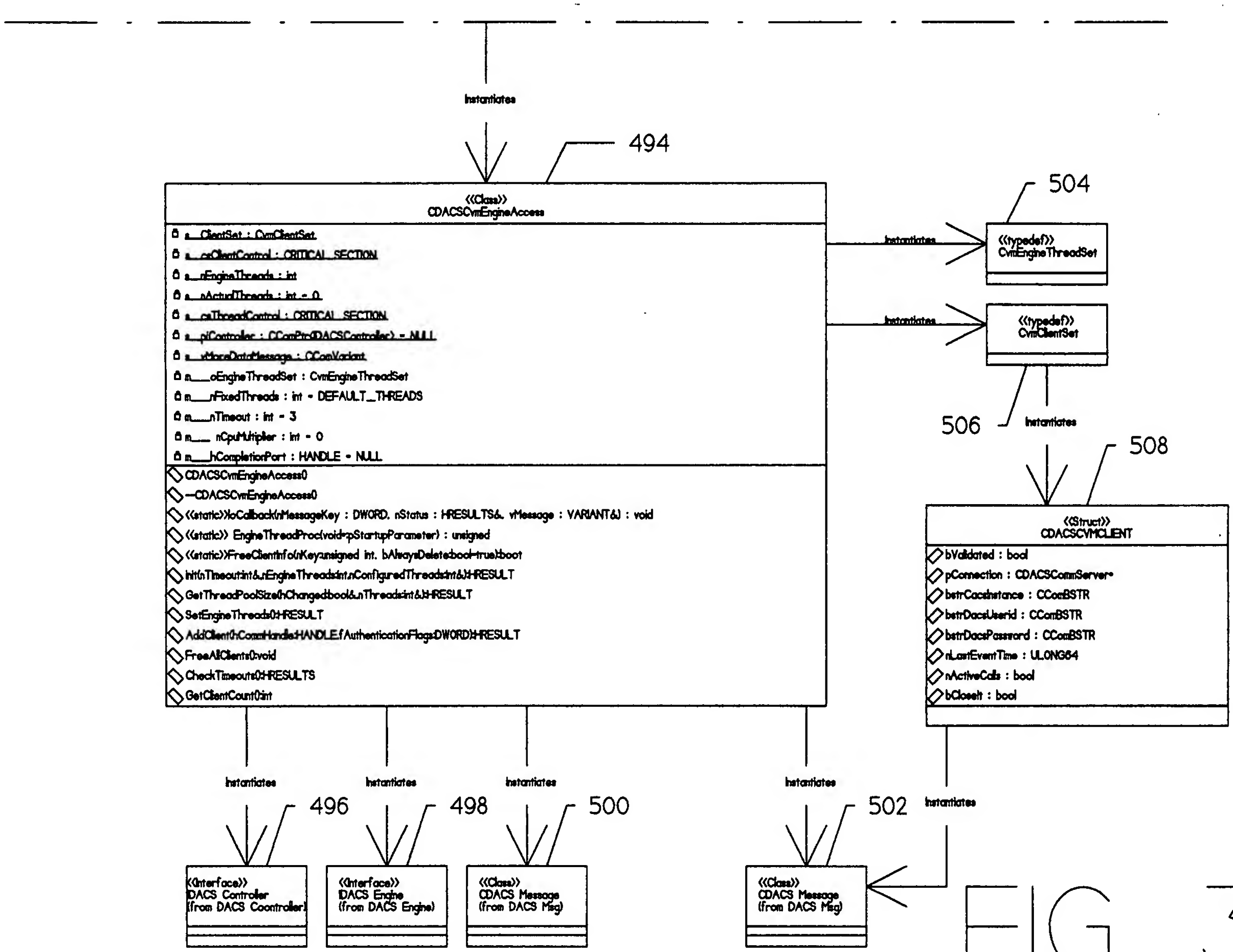
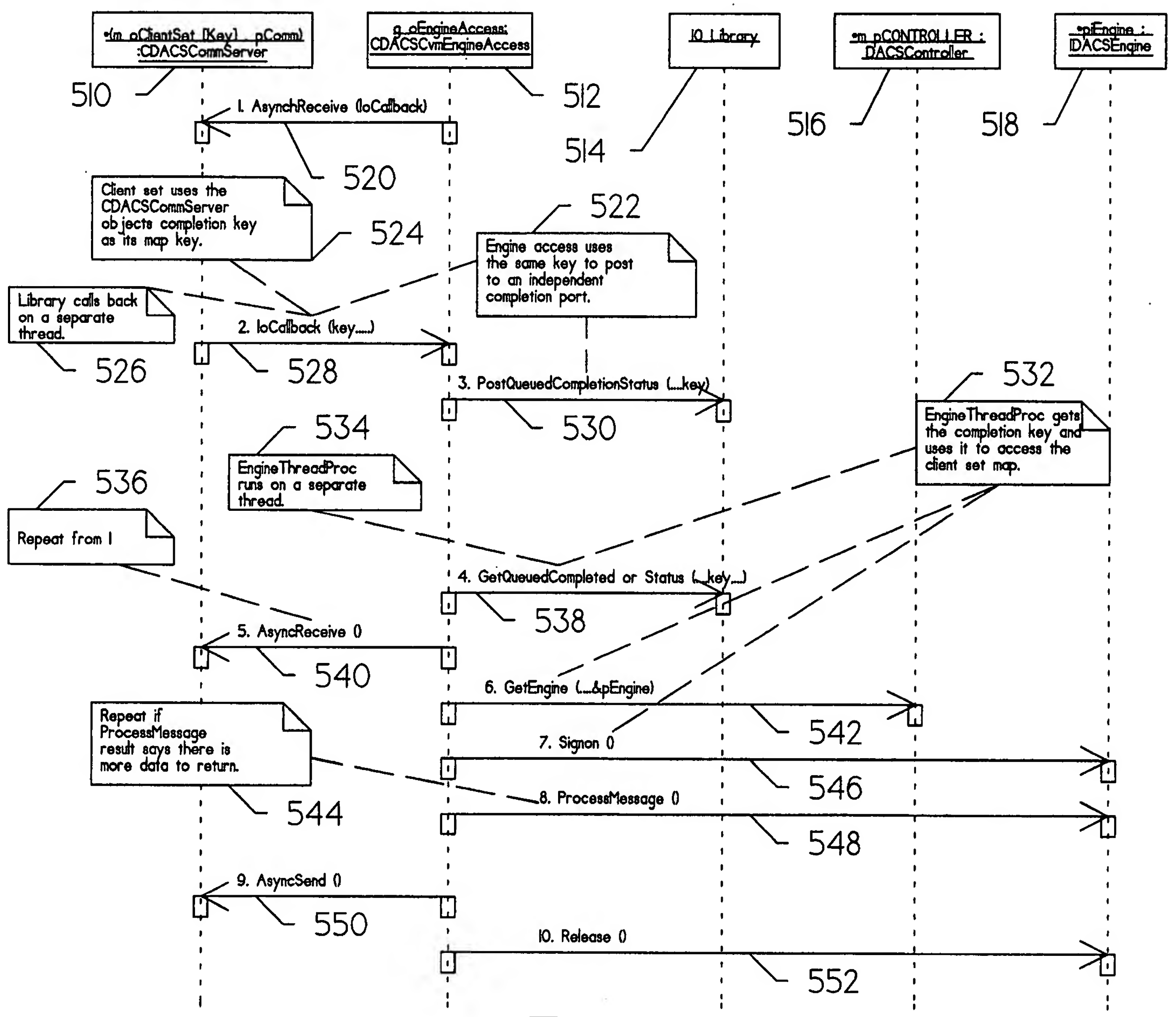

Instantiates
494
<<Class>>
CDACSCvmEngineAccess
s__ClientSet : CvmClientSet
s__csClientControl : CRITICAL_SECTION
s__nEngineThreads : int
s__nActualThreads : int = 0
s__csThreadControl : CRITICAL_SECTION
s__piController : CComPtr<IDACSController> = NULL
s__vMoreDataMessage : CComVariant
m___oEngineThreadSet : CvmEngineThreadSet
m___nFixedThreads : int = DEFAULT_THREADS
m___nTimeout : int = 3
m___ nCpuMultiplier : int = 0
m___hCompletionPort : HANDLE = NULL
CDACSCvmEngineAccess()
~CDACSCvmEngineAccess()
<<static>>IoCallback(nMessageKey : DWORD, nStatus : HRESULTS&, vMessage : VARIANT&) : void
<<static>> EngineThreadProc(void*pStartupParameter) : unsigned
<<static>>FreeClientInfo(nKey:unsigned int, bAlwaysDelete:bool=true):bool
Init(nTimeout:int&,nEngineThreads:int,nConfiguredThreads:int&):HRESULT
GetThreadPoolSize(hChanged:bool&,nThreads:int&):HRESULT
SetEngineThreads():HRESULT
AddClient(hCommHandle:HANDLE,fAuthenticationFlags:DWORD):HRESULT
FreeAllClients():void
CheckTimeouts():HRESULTS
GetClientCount():int
Instantiates
504
<<typedef>>
CvmEngineThreadSet
Instantiates
<<typedef>>
CvmClientSet
506
Instantiates
508
<<Struct>>
CDACSCVMCLIENT
bValidated : bool
pConnection : CDACSCommServer*
bstrCacheInstance : CComBSTR
bstrDacsUserid : CComBSTR
bstrDacsPassword : CComBSTR
nLastEventTime : ULONG64
nActiveCalls : bool
bCloseIt : bool
Instantiates
496
<<Interface>>
IDACS Controller
(from DACS Controller)
Instantiates
498
<<Interface>>
IDACS Engine
(from DACS Engine)
Instantiates
500
<<Class>>
CDACS Message
(from DACS Msg)
Instantiates
502
<<Class>>
CDACS Message
(from DACS Msg)
Instantiates

FIG. 3B

BEST AVAILABLE COPY



•(m_oClientSet [Key]._pComm) :CDACSCommServer
g_oEngineAccess: CDACSCvmEngineAccess
IO Library
•m_pCONTROLLER : DACSController
•piEngine : IDACSEngine
510
512
514
516
518
1. AsynchReceive (IoCallback)
520
Client set uses the CDACSCommServer objects completion key as its map key.
524
522
Engine access uses the same key to post to an independent completion port.
Library calls back on a separate thread.
526
2. IoCallback (key.....)
528
3. PostQueuedCompletionStatus (....key)
530
532
EngineThreadProc gets the completion key and uses it to access the client set map.
534
EngineThreadProc runs on a separate thread.
536
Repeat from 1
4. GetQueuedCompleted or Status (...key...)
538
5. AsyncReceive ()
540
6. GetEngine (....&pEngine)
542
Repeat if ProcessMessage result says there is more data to return.
7. Signon ()
546
8. ProcessMessage ()
544
548
9. AsyncSend ()
550
10. Release ()
552

FIG. 4

Message #	Description	Fig. 16 Reference
1	Ask to receive a message from the client. Caller is AddClient or EngineThreadProc.	520
2	DACSComm library calls IoCallback when a complete message has been received on the socket associated with this object. Multiple such calls can happen simultaneously on parallel threads for different clients.	528
3	IoCallback posts the incoming message to the engine access queue. and returns to the DACSComm library.	530
4	EngineThreadProc picks the oldest message off the queue and processes it. Multiple instances of EngineThreadProc can be doing this simultaneously on parallel threads. The client set structure for the given key provides a pointer to the CDACSCommServer object. and the information for calling the engine.	538
5	EngineThreadProc calls AsyncReceive before it calls the engine. This gives the client an opportunity to send a "Cancel" message.	540
6	Get an engine interface from the DACS Controller. using the instance name stored in the client set for this key.	542
7	Use credentials stored in the client set entry to sign onto the engine.	546
8	Marshal the client's message to the engine interface. Receive a response message in the same parameter.	548
9	Send the response back to the client. Do not ask for a callback.	550
10	Release the engine interface when there is no more data to receive from the engine and send to the client.	552

FIG. 5